DS-UA 202 Final Project Report

Name: Hongyi Zheng, Maggie Wang

Automated Detection System: Inappropriate Comment Classification System

Background:

The ADS aims to correctly classify different types of inappropriate comments like threats, obscenity, insults, and identity-based hate in comments. Compared with existing publicly available systems, this model allows users to select the type of inappropriate comments they are interested in finding (different websites may prefer different kinds of inappropriate content based on their functions and properties). By detecting and labeling inappropriate comments, the system manages to flag various types of offensive comments posted on the platform so that the administrators could choose to remove ones that violate the community standards in order to ensure that those posted comments won't hurt any subpopulations. Also, different versions of classification may have different strengths. For instance, some models have lower false-negative rates, while others have higher precisions. In this report, we will discuss how we will implement models with different strengths and how to choose between those models in terms of fairness and transparency.

Input:

The dataset is composed of comments collected from Wikipedia's talk page edits. In **`train.csv`**, we are given plenty of comments and their labels. In the **`test.csv`**, we are given some comments without labels, and the predictions of our test set could be verified in

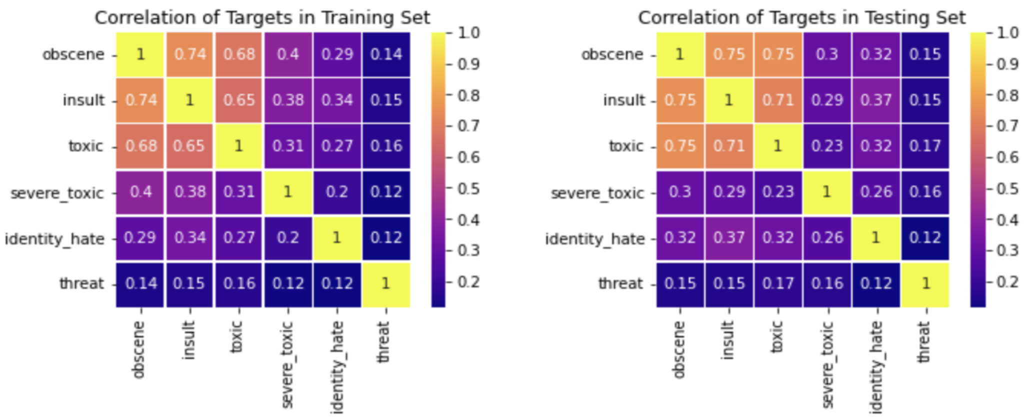
test_labels.csv, which is not visible to the author of the solution we used at the time of competition.

The input features include id and comment text. The type of feature **id** is string, and **id** is the distinct label for each user that posts comments on the website. The feature **comment_text** also belongs to the string type, which is the detailed content of the comment posted. We find that there are no missing values in this dataset, so there is no need to clean the dataset. After performing EDA, we could go straight into the data preprocessing.

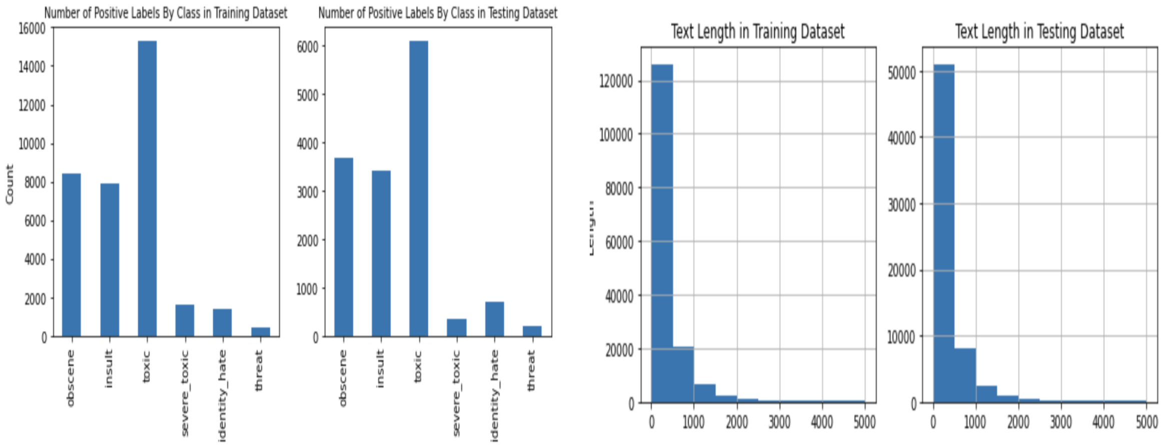
One thing worth noticing is that the datasets we are using are originally for the Kaggle competition. Therefore, only a portion of the test dataset is used for scoring, and the ground truth label for test data that are not used for scoring is not provided (marked as -1). We dropped this part of the data from the testing dataset, using the rest of the testing dataset to evaluate model performance.

In the training set, approximately 89.83% of comments have negative labels for all categories, whereas some other label combinations only account for 0.001% of the dataset, which may lead to a biased model. Hence, we plan to further evaluate the model performance later and try using upsampling/downsampling if there is any bias in this model.

In addition, since the only input feature here is the comment text, it is not possible to analyze the correlation between input features. However, since there are multiple target columns, we could compute correlations between these target columns. From the two heatmaps that indicate the correlation between each label in the training and test set below, we can discover that the correlations between the label “obscene” and label “insult”, label “obscene” and label “toxic”, and label “insult” and label “toxic” are relatively higher than others. Nevertheless, the correlations between them are not too high in general.



Then, we compare the label distribution as well as the distribution of text length separately in the training set and test set:

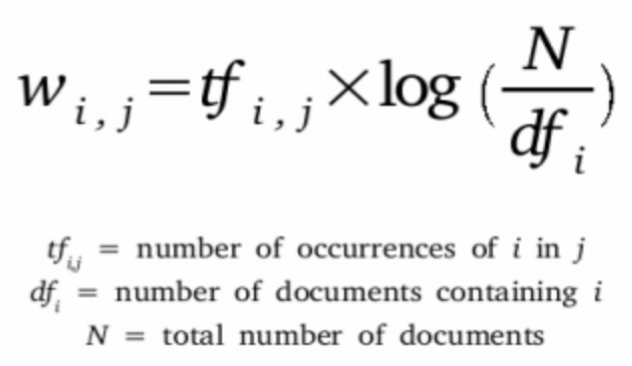


The two graphs show that the distributions of positive labels by class and text length are quite similar between the training and test sets, indicating that they may come from the same data distribution, which prevents additional bias from being introduced to the model.

Implementation and Validation:

We mainly use the solution from Rhodium Beng as the guideline for our model implementation, which involves preprocessing the input text, vectorizing the data with a TF-IDF vectorizer, and using Logistic Regression to build a multi-label classifier. Nevertheless, based on our needs, we also introduced multiple improvements compared with the original implementation. In this section, we will introduce several key components of this model as well as our improvements.

Given the input as raw comment text, a TF-IDF vectorizer is used to transform the text into vectors for further analysis. The TF-IDF is the product of two statistics, term frequency, and inverse document frequency. The TF-IDF value *wi,j* is calculated as below:



$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

$tf_{i,j}$ = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

As the formula suggests, *wi,j* increases if the number of occurrences of the word increases and decreases if the number of documents containing that word increases. Hence, TF-IDF could capture some rare words that truly contribute to comments' labels during analysis by giving a higher value to those rare words. After applying the TF-IDF transformation, we are then able to apply logistic regression models to our training data.

In the previous section, we find that approximately 89.83% of comments have negative labels for all categories in the training set, indicating the imbalance of the training set. Without techniques to solve the imbalanced data, the model might come up with poor performance in minority classes. Hence, in addition to the author's implementation, we decide to deploy the Synthetic Minority Oversampling Technique (SMOTE), which is a type of data augmentation, to fix this problem. The idea behind SMOTE is to produce a new synthetic example by picking up a random point between a minority class example and one of its nearest neighbors. Since the new synthetic examples are close to the original example in the feature space, the new examples are plausible and prevent the minority class from being under-represent. Two versions of models are trained: one version of the model is trained on the original training data, while another is trained on the training data after oversampling. The performance of the two versions of models will be compared in detail in the outcomes section.

Also from the input section, we have observed that the correlation between each label is not that high. As a result, we choose to use binary relevance, which treats each label as a separate single classification problem. More specifically, for each version of the models, we run six logistic regressions, one for each target label, and report the accuracy of every logistic regression classifier. Therefore, there will be 6 × 2 = 12 logistic regression models in total. The utilization of binary relevance also makes fairness among all labels (e.g. the system performs similarly across different kinds of inappropriate comments) possible since if we use other techniques such as classifier chain, the model would likely perform better on target variables in the latter part of the chain.

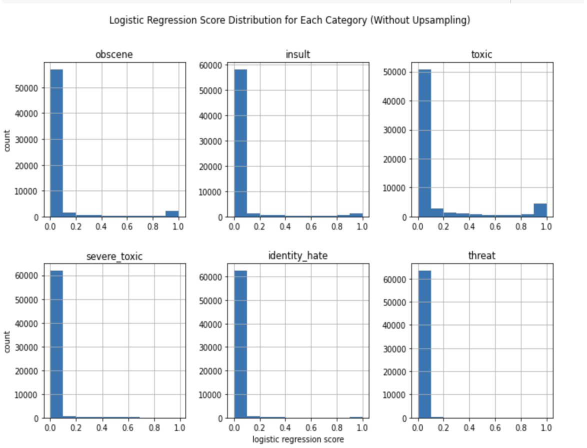
Another important improvement we have done to the model is that we introduced cross-validation and tuned the penalty term of the logistic regression models by replacing **`LogisticRegression()`** with **`LogisticRegressionCV()`**. In this way, we are able to prevent overfitting to boost the model performance.

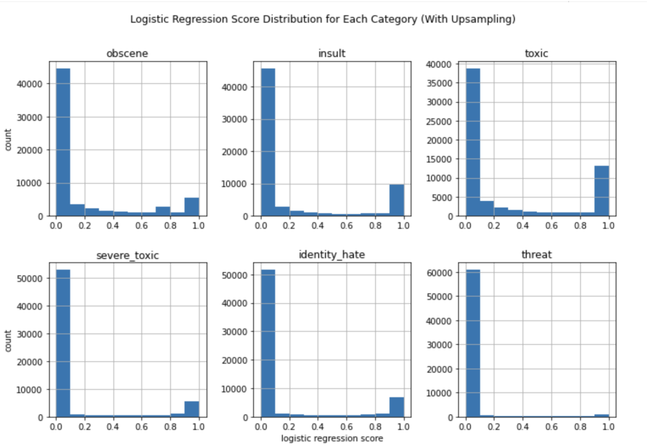
We also upgraded the text cleaning function that is used prior to applying the TF-IDF vectorizer, which may help improve the performance of the logistic regression by reducing noises in texts.

Output:

The output of this system is the probability that indicates the comments being positive for each of the six inappropriate types: if the probability of a certain comment under an inappropriate type is greater than 0.5, then this comment will be classified as positive. For example, if the values of a comment under toxic and insult are greater than 0.5 while the values under other inappropriate types are smaller than 0.5, that indicates this comment is toxic and insulting but not severely toxic, obscene, threat, or identity hate. To complete the final classification, we transform the probability into binary labels for further use: mark as negative (value 0) for probability smaller than 0.5 and mark as positive (value 1) for a probability greater than 0.5. Note that this is different from multi-class classification as a comment may have more than one label. Thus, multiple separate logistic regression classifiers are used to predict each label for every comment.

In addition, for each inappropriate comment type, we visualize the distribution of the logistic regression score, which is the predicted probability of the comment being positive for a certain inappropriate comment type.





We could observe that regardless of upsampling the train data or not, almost all logistic regression scores for every class are very close to 0 and 1, indicating that the model is quite confident in whether the test is positive for each category or not. Nevertheless, the model after upsampling tends to classify more testing data as positive for each target variable.

	Without Upsampling	**With Upsampling**
Obscene	0.9817	0.9649
Insult	0.9742	0.9520
Toxic	0.9624	0.9377

Severe Toxic	0.9907	0.9823
Identity Hate	0.9930	0.9798
Threat	0.9976	0.9992

Table 1 Training Accuracy On Different Target Variables

As presented in the table, all accuracy values on the training set are higher than 96%, which suggests that the performance of these models on the training set is satisfactory.

Outcomes:

We select several metrics to evaluate the model performance on the test set, including accuracy, recall, precision, false-negative rate, and false-positive rate. The tables below show the metrics of the models trained on original data and upsampled data.

	Obscene	**Insult**	**Toxic**	**Severe Toxic**	**Identity Hate**	**Threat**
Accuracy	0.9654	0.9636	0.9287	0.9932	0.9910	0.9962
Recall	0.6681	0.5518	0.7534	0.2752	0.3666	0.2938
Precision	0.7140	0.7043	0.6001	0.3769	0.6779	0.4026
FNR	0.3319	0.4482	0.2466	0.7248	0.6334	0.7062
FPR	0.0164	0.0131	0.0528	0.0026	0.0020	0.0014

Table 2 Model Metrics Without Upsampling

	Obscene	**Insult**	**Toxic**	**Severe Toxic**	**Identity Hate**	**Threat**
Accuracy	0.8734	0.8539	0.8263	0.9127	0.9070	0.9910
Recall	0.8648	0.8369	0.9095	0.6185	0.6798	0.6066
Precision	0.2957	0.2461	0.3440	0.0400	0.0780	0.2071
FNR	0.1352	0.1631	0.0905	0.3815	0.3202	0.3934

FPR	0.1261	0.1451	0.1825	0.0856	0.0905	0.0077

Table 3 Model Metrics With Upsampling

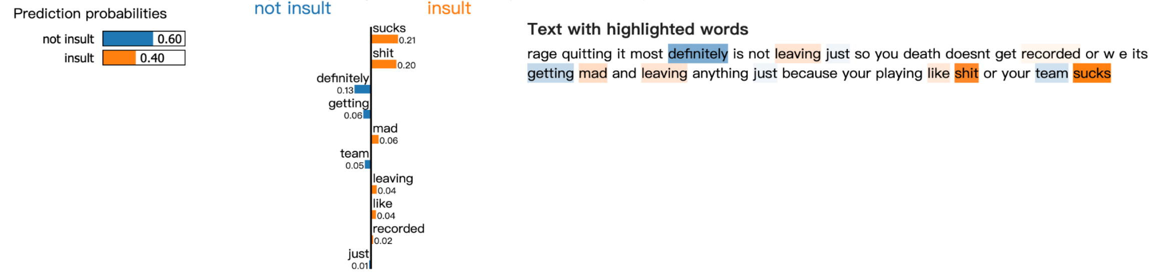
Comparing the metric values between two tables, we could discover that recall increases and false-negative rate drops significantly when adopting upsampling. At the same time, the accuracy and precision drop a little. The fluctuation of these values implies a tradeoff between different metrics. In this ADS, the low precision arises due to the small number of true positive examples resulting from imbalanced data. Hence, we may prefer the models with upsampling for higher recall and lower false-negative rates instead of focusing on precision.

To measure fairness among different labels, all metrics of the six logistic regression models are reported. We observe that the false-negative rate of threatening comments is higher than other types of comments. In addition, the false-positive rate of toxic comments is higher than other types of comments.

Next, we use the LIME interpreter to see how these logistic regression models make the decision. Also, we compare two versions of models (with or without upsampling) to see which would make a better decision. A set of representative examples are carefully chosen and visualized to better illustrate how those two versions of models work. The graph attached below is an obscene comment. While the model trained without upsampling data labels it as not obscene, the model trained with upsampling data successfully predicts the right label.

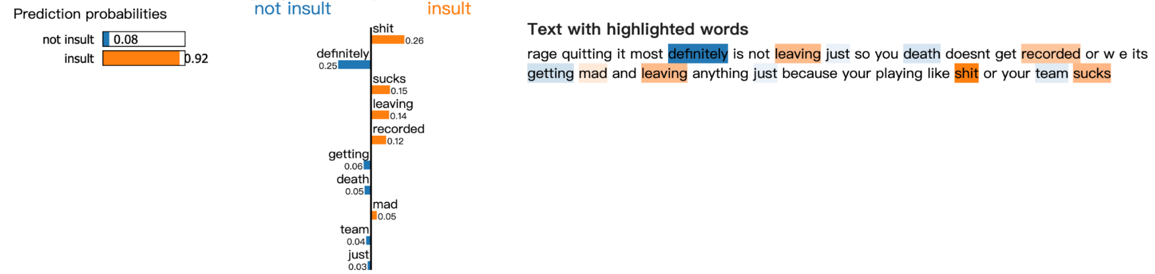
We observe that both versions of models can correctly identify offensive words, and there are cases for each version of the model to outperform another. In conclusion, the decision-making process of these models is generally reasonable, and we could trust those classification models. Platforms could choose among these two versions of models based on desired censorship strength: if a platform wants to filter as many inappropriate comments as possible to protect its users, then the model trained with upsampled data is a better choice. In contrast, if a platform wants to encourage freedom of speech and only wishes to eliminate a few extremely unacceptable comments, then the model trained with the original data is a better choice.

```
Actual: insult
Prediction of model trained with original training data: not insult (misclassification)
```



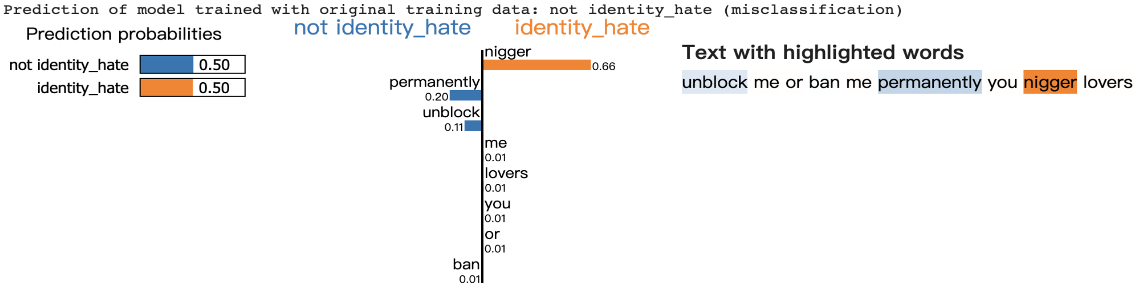
rage quitting it most definitely is not leaving just so you death doesnt get recorded or w e its getting mad and leaving anything just because your playing like shit or your team sucks

```
Prediction of model trained with upsampled training data: insult
```



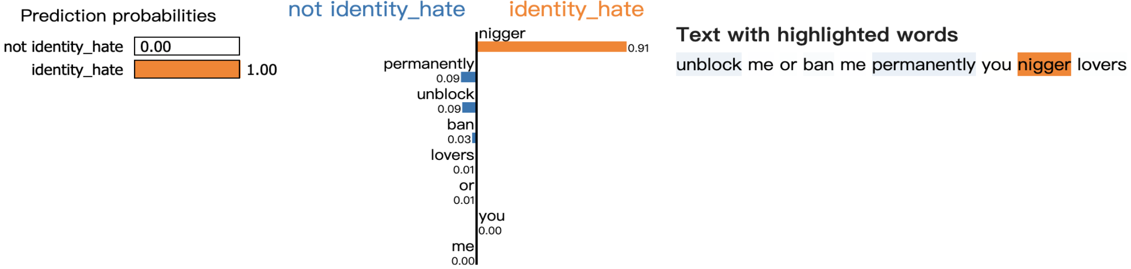
rage quitting it most definitely is not leaving just so you death doesnt get recorded or w e its getting mad and leaving anything just because your playing like shit or your team sucks

```
Actual: identity_hate
Prediction of model trained with original training data: not identity_hate (misclassification)
```



unblock me or ban me permanently you nigger lovers

```
Prediction of model trained with upsampled training data: identity_hate
```



unblock me or ban me permanently you nigger lovers

Summary:

The data used to train the inappropriate comment classification system is appropriate, but not sufficient. In our case, because of the imbalance of the data, upsampling techniques need to be employed to improve several metrics (at the cost of making some other metrics worse).

Generally speaking, the models we used to implement the ADS manage to yield acceptable performance based on those metrics, including accuracy, recall, precision, false-positive rate, and false-negative rate. Nevertheless, if more data, especially those with positive labels could be provided, then large improvements in the performance of this ADS could be expected.

The stakeholders of this ADS involve the users and the platforms. Some users, especially those who post comments on the website, might want an ADS with a lower false-positive rate. A lower false-positive rate means that the comments are less likely to be identified as toxic by mistake, so the users can express their thoughts more freely. The platforms and some users (especially the ones that belong to certain demographic groups that are more subject to discrimination) might prefer a lower false-negative rate as it ensures that almost all inappropriate comments are filtered, thus creating a peaceful atmosphere on the platform.

In terms of accuracy, our models are without any doubt very powerful. However, this metric might not be that meaningful since the testing data is highly imbalanced. In terms of fairness, there are still many improvements needed for our models. For instance, compared with other types of inappropriate comments, both versions of models have a relatively high false-negative rate for identifying severe toxic, identity hates, and threatening comments, which implies that it might not be able to provide sufficient protection to users from minority and vulnerable groups.

Finally, as the purpose of this ADS is to filter inappropriate comments, which will have positive impacts once deployed, it should be adopted in the public sector and industry given that its performance metrics are satisfactory. Nevertheless, by far this ADS is still quite immature and needs further training with more complex models such as time-series based models and more comprehensive data, especially balanced data, to minimize misclassifications.